POLYTECHNIQUE MONTRÉAL

LE GÉNIE EN PREMIÈRE CLASSE

Guide TP4

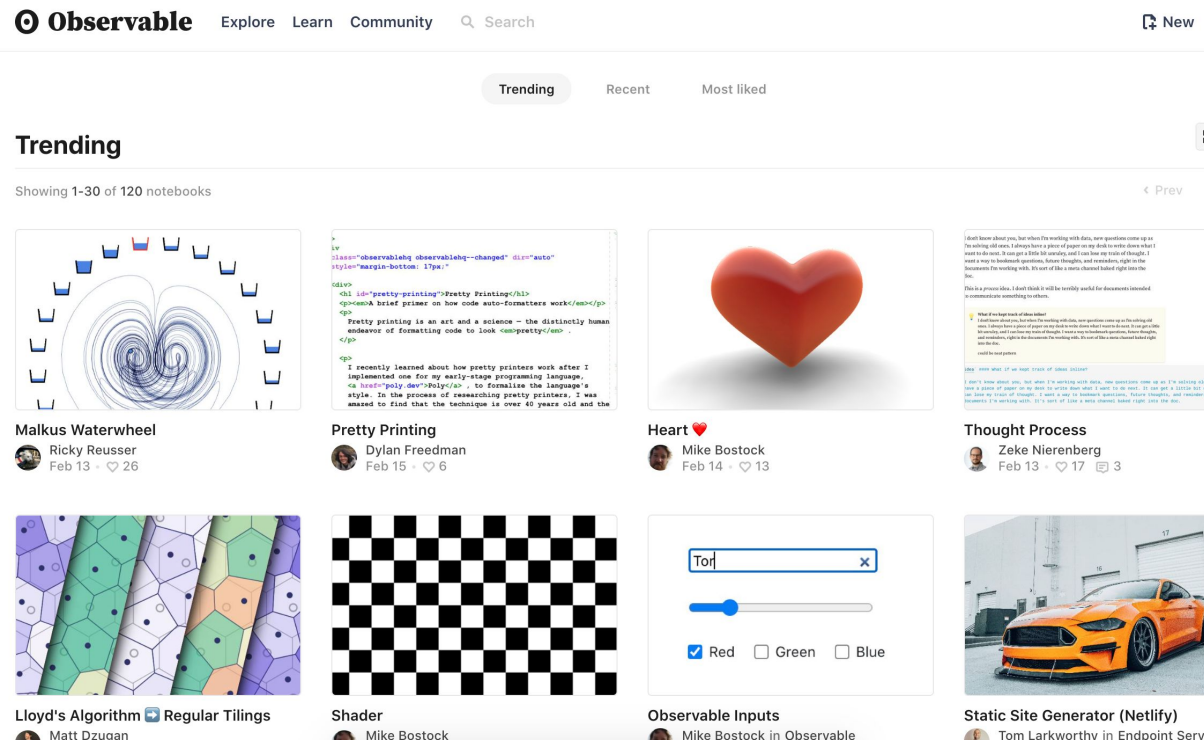
INF8808 | Summer 2022

Version JavaScript

Observable notebook

Cahier *Observable*

- *Observable* is a website which allows you to create JavaScript notebooks
- It's often used to help create data visualizations and with D3
  - *Observable* was founded by the creator of D3



Observable

Steps to fill the notebook

1. Create an account https://observablehq.com/
2. Go on the link for the TP
   - https://observablehq.com/d/836639e52760f996
3. Use the “fork” button to create your own copy of the notebook
4. Fill the functions directly in the notebook

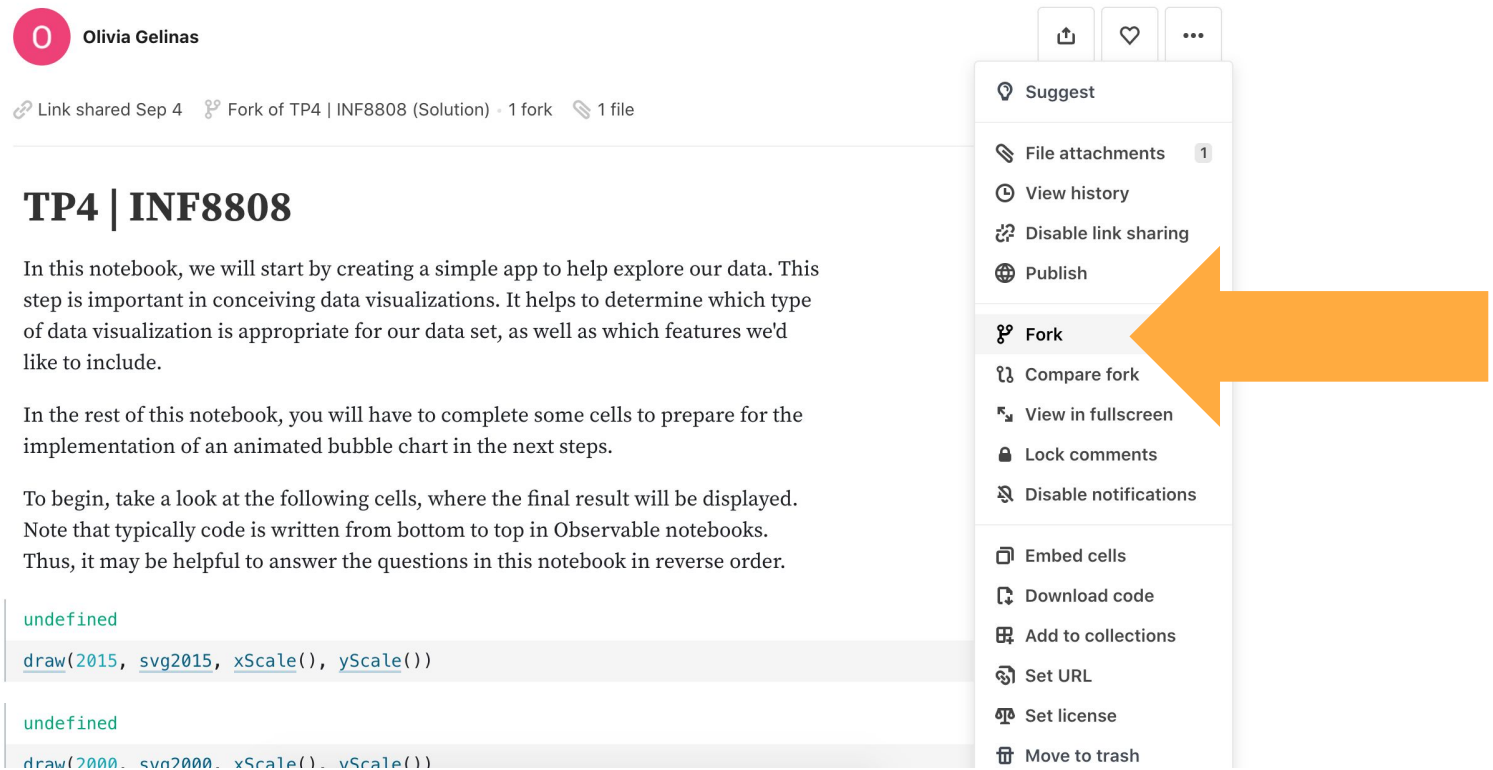
The data is available directly in the notebook like so :

```
countries = ▸ Object {2000: Array(174), 2015: Array(174)}
```

```
countries = FileAttachment("countriesData.json").json()
```

Observable

« Fork »

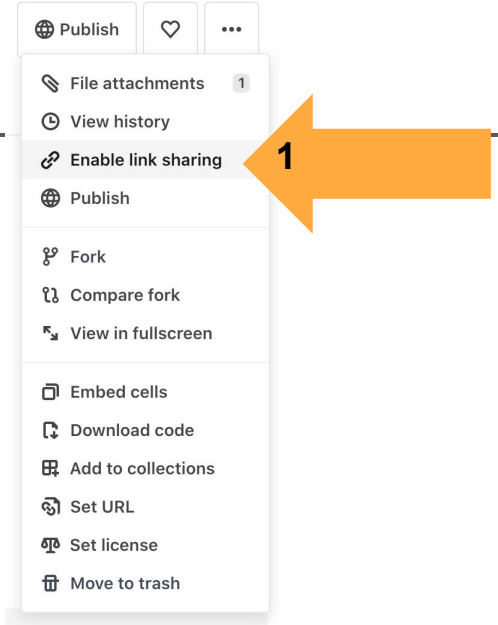


Observable

Submission

1. Make sure to enable “link sharing”

2. Copy paste the URL in the file *.observableInfo* in the TP code on Moodle



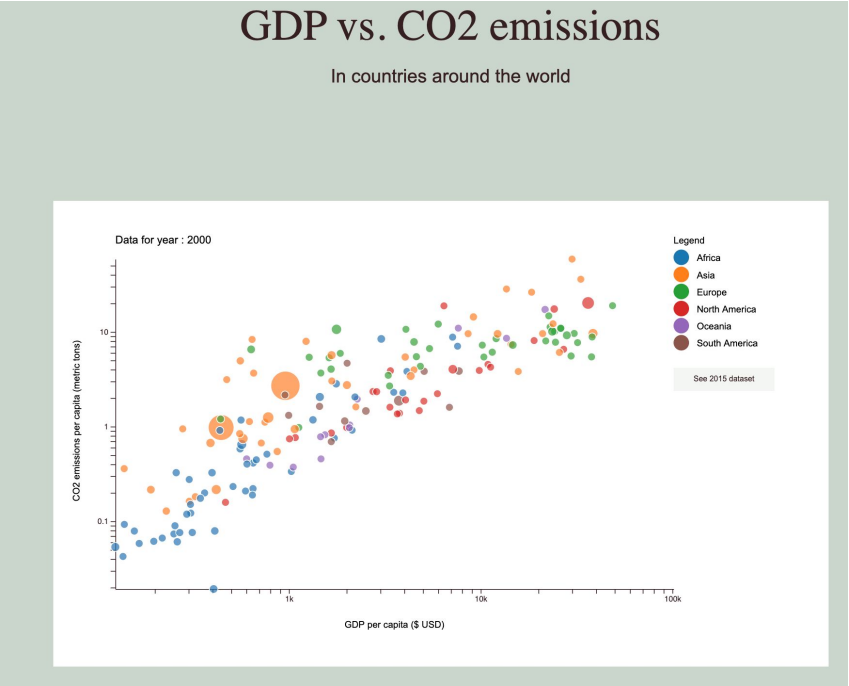
Important notes :

DO NOT MAKE NOTEBOOK PUBLIC.

MAKE SURE NOTEBOOK IS ACCESSIBLE TO T.A.’s BY TESTING THE LINK YOU SUBMIT.

Goals

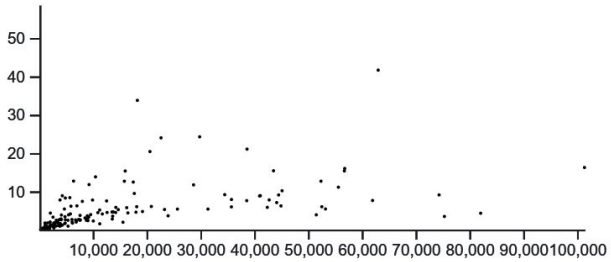
- Create an animated bubble chart from JSON data.



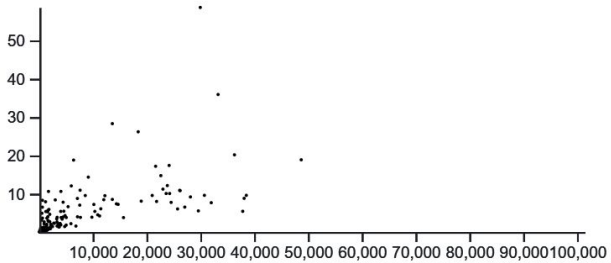
Goals

- Before implementing the code, you will create a simplified version in the *Observable* notebook.

- Link : https://observablehq.com/d/836639e52760f996




```
draw(2015, svg2015, xScale(), yScale())
```



Data

countriesData.json

- Data about various countries of the world over two years
- File : `src/assets/data/countriesData.json`
- Columns :
  - **Country Name**
  - **GDP**, country's GDP in USD
  - **CO2**, country's CO2 emissions in metric tonnes
  - **Population**
  - **Continent**

Data

Example

```
{
  "2015":[
     {
        "Country Name":"Albania",
        "GDP":3952.8012152447,
        "CO2":1.6026480342,
        "Population":2880703,
        "Continent":"Europe"
     },
     { … },
     …
    ],
   “2000”: [ … ]
}
```

Data exploration

But *: Explore the data by filling the Observable notebook*

In the notebook on *Observable* :

1. Fill the sections from bottom to top to trace the axes and points
   - The convention is to write code bottom to top in *Observable*

Question 3 :

```
xScale = ƒ()
```

```
xScale = function setXScale () {
  // TODO : Define the linear scale in x for the scatter plot
}
```

Question 2 :

```
yScale = ƒ()
```

```
yScale = function setYScale () {
  // TODO : Define the linear scale in y for the scatter plot
}
```

Question 1 :

```
draw = ƒ(year, g, xScale, yScale)
```

```
function draw (year, g, xScale, yScale) {
  // TODO : Draw scatter plot, including axes and circular markers
}
```

Scales

But : *Generate the scales used for the bubble chart*

- Dans le fichier `scales.js` :

1. **setRadiusScale**, creates a linear scale for the radius
2. **setColorScale**
3. **setXScale**, creates a log scale for the x axis
4. **setYScale**, creates a log scale for the y axis

Animated bubble chart

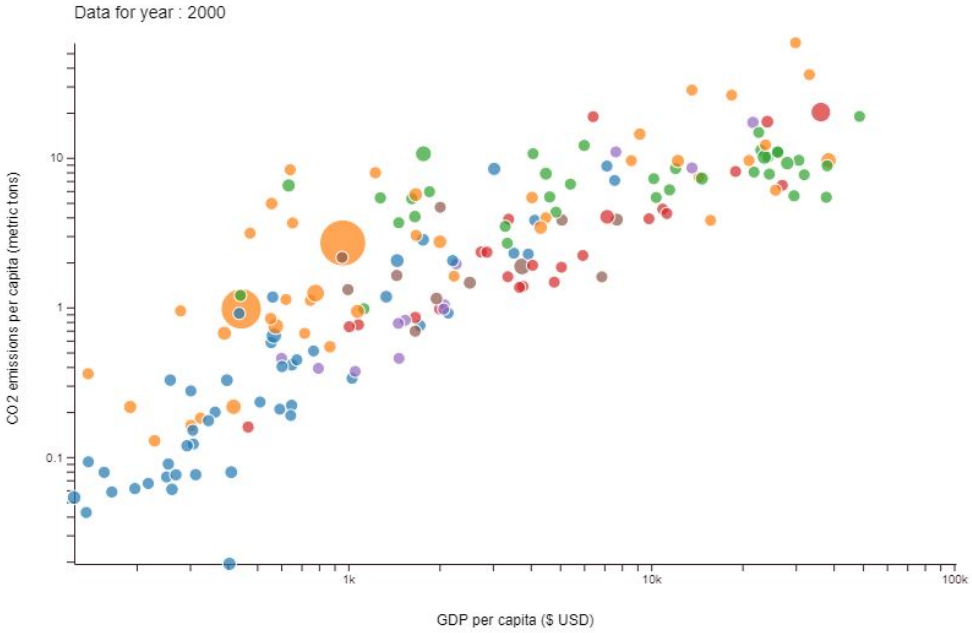
But

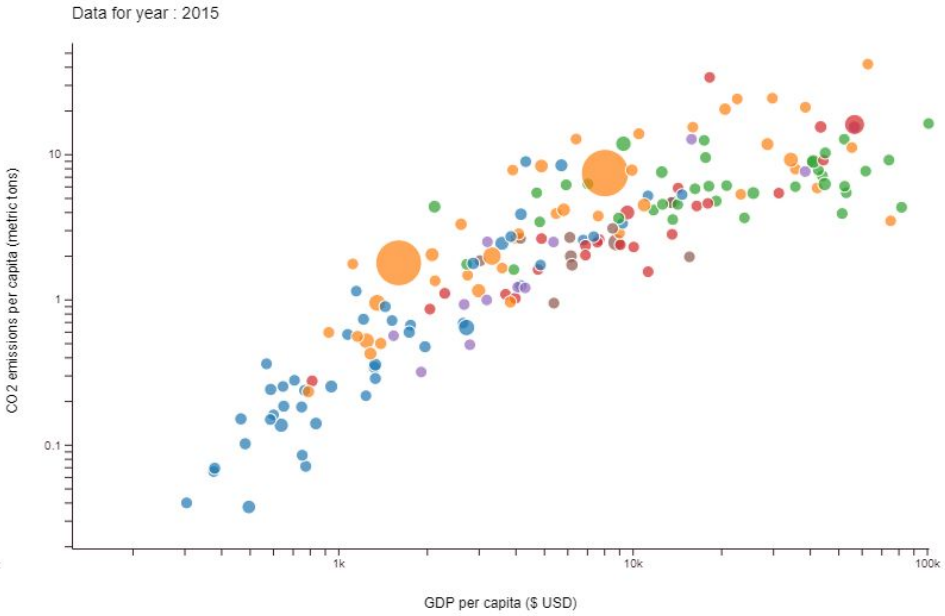
Trace the bubble chart, where:

- The position of the center of the circles in **x** corresponds to the circle's **GDP**
- The position of the center of the circles in **y** corresponds to the circle's **CO2**
- The **color** of each circle corresponds to its **continent**
- The **radius** of each circle corresponds to its **population**
- The axes are labelled like in the TP subject
- The circle opacity is **70%** when the circle **is not hovered**
- The circle opacity is **100%** when the circle **is hovered**
- When the data is updated the circles move to their new position with a **D3 transition**

Animated bubble chart

Goal : *Animate between the two views of the bubble chart*





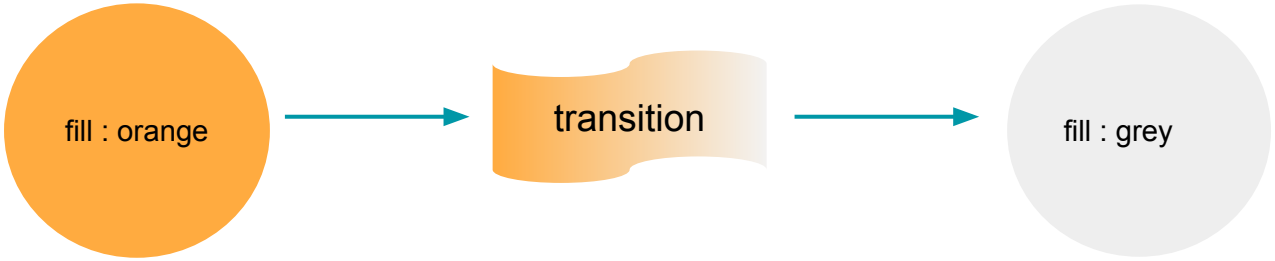
Animated bubble chart

Transition D3

- To animate, use **d3.transition()**

```
d3.selectAll(...)
.attr(...)//attribute before transition (optional)
.transition()
.delay(X).duration(Y).ease(Z)//optional config of transition
.attr(...) //attribute after transition (optional)
```

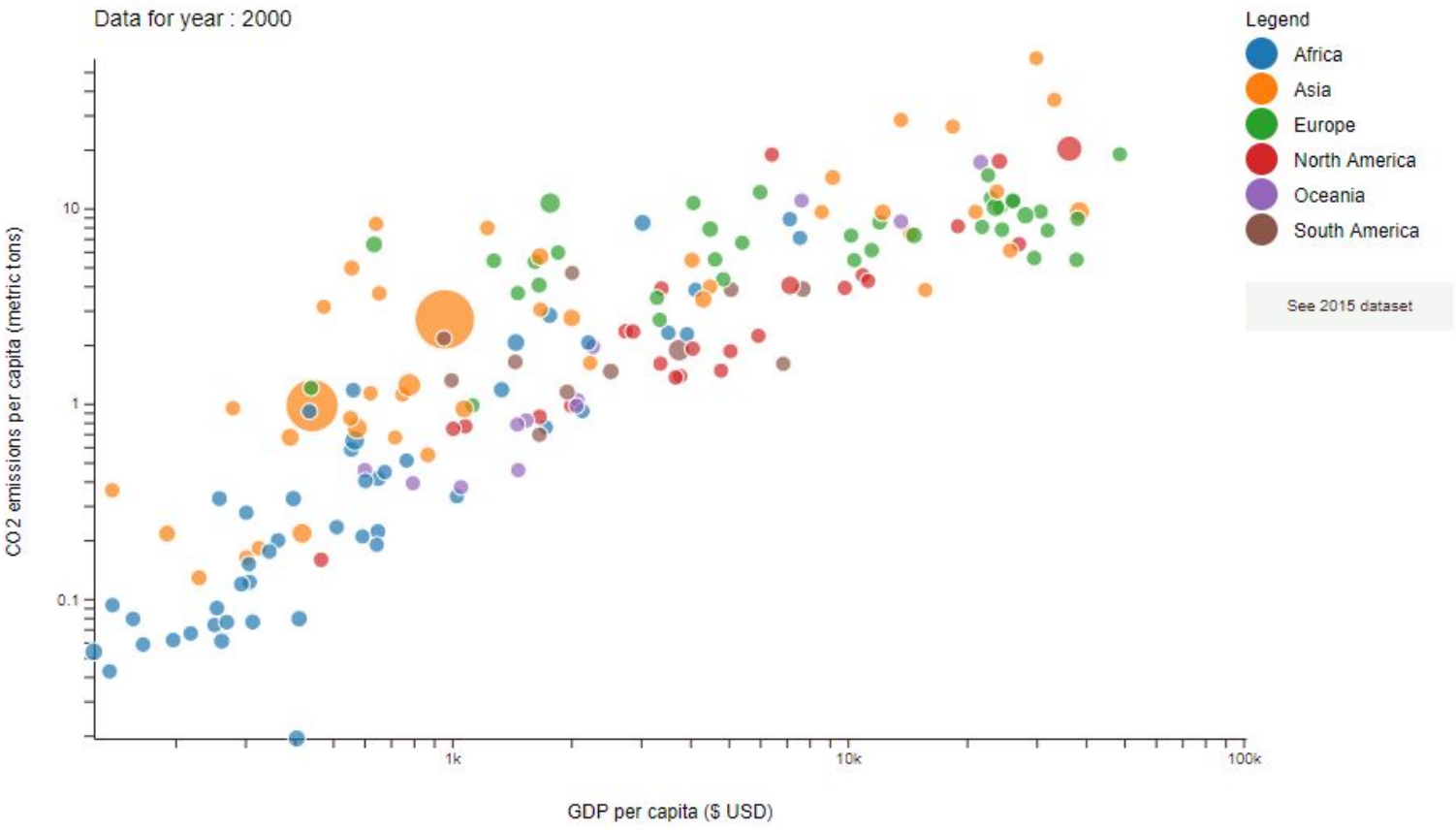
<u>Exemple :</u>



<circle> selection before

<circle> selection after

Legend


Data for year : 2000
Legend
Africa
Asia
Europe
North America
Oceania
South America
See 2015 dataset
CO2 emissions per capita (metric tons)
10
1
0.1
1k
10k
100k
GDP per capita ($ USD)

Légende

The legend indicates which continent each color in the graph corresponds to.

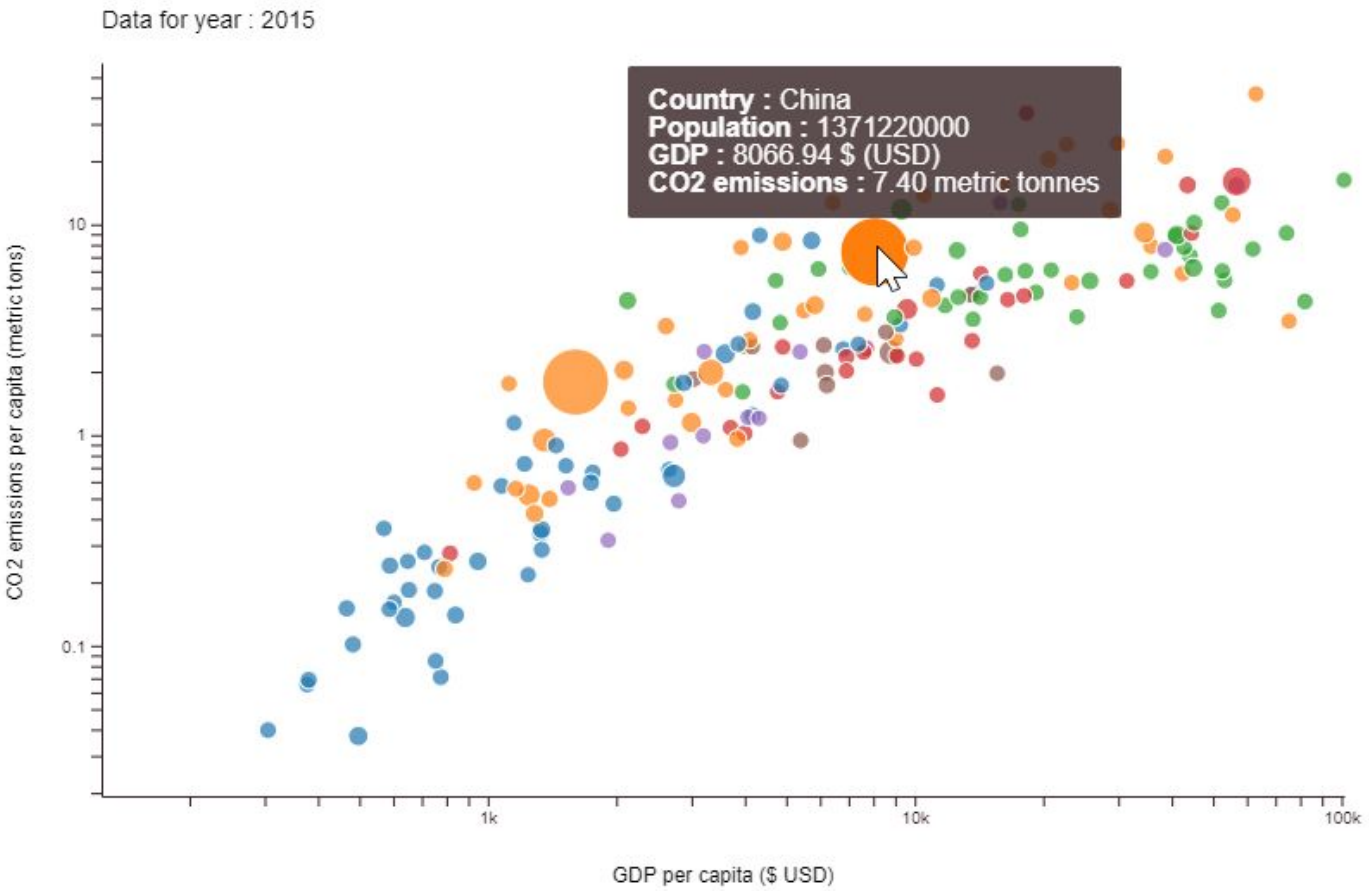
- Use the imported library in `legend.js` :

```
import d3Legend from 'd3-svg-legend'

/**
 * Draws the legend.
 *
 * @param {*} colorScale The color scale to use
 * @param {*} g The d3 Selection of the graph's g SVG element
 * @param {number} width The width of the graph, used to place the legend
 */
export function drawLegend (colorScale, g, width) {
  // TODO : Draw the legend using d3Legend
  // For help, see : https://d3-legend.susielu.com/
}
```

Tooltip

Example



Tooltip

- Fill the code in `tooltip.js`.

```
/**
 * Defines the contents of the tooltip. See CSS for tooltip styling. The tooltip
 * features the country name, population, GDP, and CO2 emissions, preceded
 * by a label and followed by units where applicable.
 *
 * @param {object} d The data associated to the hovered element
 * @returns {string} The tooltip contents
 */
export function getContents (d) {
  // TODO : Generate tooltip contents
  return ''
}
```

Notes TP4

Observable

- In the notebook from *Observable*, if you see nothing appearing, make sure the function *draw()* returns an HTML node
  - See : `.node()`

- If you still see nothing, try disabling browser extensions that could block JavaScript code from executing
  - Ex : ***Ad Block, NoScript, etc.***

Notes TP4

Échelles

- Reminder : A scale in d3 is used to convert values from the *domain* into values from the *range* according to the defined transformation
- For example, see : **scaleLinear**, **scaleOrdinal**, **scaleLog** from D3
- The exists color schemes that can be used to configure color scales, such as **d3.schemeCategory10**
- To create a scale with this *scheme* as a *range* :
  - `color = d3.scaleOrdinal(d3.schemeCategory10)`

d3.**schemeCategory10** <>

An array of ten categorical colors represented as RGB hexadecimal strings.

Notes TP4

Legend

- The library documentation for more help as needed : https://d3-legend.susielu.com/

- In particular, notice how to make legends with custom shapes (circle, rectangle, etc.)

Due date

June 5th 2021, 11:59PM